IN THE CLAIMS

Please amend the claims as follows:

1. (Previously Presented) A method of compiling computer code including power-down instructions to reduce power consumption during execution of the code while satisfying user-specified real-time performance constraints on a microprocessor, comprising:

identifying one or more potential locations in the computer code where the power-down instructions can be inserted;

selecting locations to insert the power-down instructions from the identified potential locations in the code based on reducing power consumption and satisfying user-specified real-time performance constraints; and

inserting the power-down instructions in the selected locations to reduce the power consumption during the execution of the code while satisfying user-specified real-time performance constraints.

2. (Original) The method of claim 1, wherein the code is written for a microprocessor having distinct functional units.

3. (Original) The method of claim 2, wherein identifying potential locations comprises:

identifying potential locations based on the functional units not being used in the potential locations, wherein the functional units not being used are determined based on functional unit usage transfer functions at each of the potential locations as specified in standard monotone data-flow frameworks.

4. (Original) The method of claim 3, wherein identifying potential locations is accomplished by statically analyzing processor cycles prior to executing the code.

5. (Original) The method of claim 4, wherein statically analyzing processor cycles is accomplished by statically analyzing the text in the code for the functional units not being used prior to executing the code.

6. (Original) The method of claim 3, wherein each of the power-down instructions comprise:

a first power-down instruction operable to reduce power to all of the at least one functional unit, such that the functional unit is placed in a low state of readiness and a second power-down instruction operable to reduce power to only a part of the at least one functional unit, such that the functional unit is placed in an intermediate state of readiness.

7. (Previously Presented) The method of claim 1, wherein selecting identified potential locations on the computer code based on satisfying the user-specified real-time performance constraints, comprise:

executing the code to generate power-profiling information associated with each of the identified potential locations;

executing the code to generate execution path-profiling information associated with each of the identified potential locations;

assigning a weight factor to each of the identified potential locations based on the generated power-profiling and path-profiling information; and

selecting the locations to insert the power-down instruction from the identified locations based on the assigned weight factors and the user-specified real-time performance constraints.

8. (Original) The method of claim 7, wherein executing the code to generate path-profiling information to each of the identified potential locations further comprises:

generating execution probability of each of the identified potential locations based on the generated path-profiling information.

9. (Original) The method of claim 8, wherein assigning the weight factor comprises:

extracting potential energy savings for each of the identified potential locations using the generated power profile analysis information; and

assigning the weight factor to each of the identified potential locations based on the extracted potential energy savings and the generated execution probability.

10. (Original) The method of claim 9, wherein assigning the weight factor further comprises:

executing the code to assign a first weight factor based on the extracted potential energy savings to each of the identified potential locations;

executing the code to assign a second weight factor based on execution probability at each of the identified potential locations;

computing a product of the first and second weight factors for each of the identified potential locations;

calculating the weight factor for each of the identified potential locations based on the computed product of the first and second weight factors; and

assigning the calculated weight factor to each of the identified potential locations.

11. (Original) The method of claim 1, wherein user-specified real-time constraints comprise:

the number of power-down instructions that can be inserted in an execution path, including one or more identified potential locations.

12. (Previously Presented) The method of claim 11, wherein user-specified real-time performance constraints comprise:

the number of additional cycles of execution time the user is willing to incur due to an insertion of the power-down instruction at each of the identified potential locations.

13. (Original) The method of claim 11, further comprising:

inserting power-up instruction in the code to restore at least one functional unit to a ready state powered-down by the inserted power-down instructions.

14. (Previously Presented) A computer-readable medium having computer-executable instructions for reducing power consumption while running a computer program, comprising:

identifying one or more potential locations in the computer program where power-down instructions can be inserted;

selecting locations to insert the power-down instructions from the identified potential locations in the program based on satisfying user-specified real-time performance constraints; and

inserting the power-down instructions in the selected locations to reduce power consumption while running the computer program while satisfying the user-specified real-time performance constraints.

15. (Original) The medium of claim 14, wherein the code is written for a microprocessor including distinct functional units.

16. (Original) The medium of claim 14, wherein identifying potential locations comprises:

identifying the potential locations based on the functional units not being used in the potential locations, wherein the functional units not being used are determined based on functional unit usage transfer functions at each of the potential locations as specified in standard monotone data-flow frameworks.

17. (Original) The medium of claim 16, wherein identifying potential locations is accomplished by statically analyzing processor cycles prior to running the program.

18. (Previously Presented) The medium of claim 14, wherein selecting the identified potential locations on the computer program based on satisfying the user-specified real-time constraints, comprise:

running the computer program to generate power-profiling information associated with each of the identified potential locations;

running the computer program to generate execution path-profiling information associated with each of the identified potential locations;

assigning a weight factor to each of the identified potential locations based on the generated power-profiling and path-profiling information; and

selecting the locations to insert the power-down instructions from the identified locations based on the assigned weight factors and the user-specified real-time performance constraints.

19. (Original) The medium of claim 18, wherein running the program to generate path-profiling information to each of the identified potential locations further comprises:

generating running probability of each of the identified potential locations based on the generated path-profiling information.

20. (Original) The medium of claim 19, wherein assigning the weight factor comprises:

extracting potential energy savings for each of the identified potential locations using the generated power profile analysis information; and

assigning the weight factor to each of the identified potential locations based on the extracted potential energy savings and the generated running probability.

21. (Original) The medium of claim 20, wherein assigning the weight factor further comprises:

running the program to assign a first weight factor based on the extracted potential energy savings to each of the identified potential locations;

running the program to assign a second weight factor based on execution probability at each of the identified potential locations;

computing a product of the first and second weight factors for each of the identified potential locations;

calculating the weight factor for each of the identified potential locations based on the computed product of the first and second weight factors; and

assigning the calculated weight factor to each of the identified potential locations.

22. (Previously Presented) The medium of claim 14, wherein user-specified real-time performance constraints comprise:

the number of power-down instructions that can be inserted in a running path including one or more identified potential locations.

23. (Original) The medium of claim 22, further comprising:

inserting power-up instructions in the program to restore at least one functional unit to a ready state powered-down by the inserted power-down instructions.

24. (Previously Presented) A computer system for reducing power consumption during execution of computer code, comprising:

a storage device;

an output device; and

a processor programmed to repeatedly perform a method, comprising:

identifying one or more potential locations in the computer code where power-down instructions can be inserted;

selecting locations to insert the power-down instructions from the identified potential locations in the code based on satisfying user-specified real-time performance constraints; and

inserting the power-down instructions in the selected locations to reduce power consumption during the execution of the code while satisfying the user-specified real-time performance constraints.

25. (Original) The system of claim 24, wherein the code is written for a microprocessor including distinct functional units.

26. (Original) The system of claim 24, wherein identifying the potential locations comprises:

identifying the potential locations based on the functional units not being used in the potential locations, wherein the functional units not being used are determined based on functional unit usage transfer functions at each of the potential locations as specified in standard monotone data-flow frameworks.

27. (Original) The system of claim 26, wherein identifying the potential locations is accomplished by statically analyzing processor cycles prior to executing the code.

28. (Previously Presented) The system of claim 24, wherein selecting the identified potential locations on the computer code based on satisfying the user-specified real-time performance constraints, comprises:

executing the code to generate power-profiling information associated with each of the identified potential locations;

executing the code to generate execution path-profiling information associated with each of the identified potential locations;

assigning a weight factor to each of the identified potential locations based on the generated power-profiling and path-profiling information; and

selecting the locations to insert the power-down instruction from the identified locations based on the assigned weight factors and the user-specified real-time performance constraints.

29. (Original) The system of claim 28, wherein executing the code to generate path-profiling information to each of the identified potential locations further comprises:

generating execution probability of each of the identified potential locations based on the generated path-profiling information.

30. (Original) The system of claim 29, wherein assigning the weight factor comprises:

extracting potential energy savings for each of the identified potential locations using the generated power profile analysis information; and

assigning the weight factor to each of the identified potential locations based on the extracted potential energy savings and the generated execution probability.

31. (Original) The system of claim 30, wherein assigning the weight factor further comprises:

executing the code to assign a first weight factor based on the extracted potential energy savings to each of the identified potential locations;

executing the code to assign a second weight factor based on execution probability to each of the identified potential locations;

computing a product of the first and second weight factors for each of the identified potential locations;

calculating the weight factor for each of the identified potential locations based on the computed product of the first and second weight factors; and

assigning the calculated weight factor to each of the identified potential locations.

32. (Previously Presented) The system of claim 24, wherein user-specified real-time performance constraints comprise:

the number of power-down instructions that can be inserted in an execution path including one or more identified potential locations.

33. (Original) The system of claim 32, further comprising:

inserting power-up instructions in the code to restore at least one functional unit to a ready state powered-down by the inserted power-down instructions.

34. (Previously Presented) A computer-readable medium having a computer program including instructions for causing a computer to perform a method of selectively controlling power to different functional units of the computer, the instructions comprising:

power-down instructions inserted in the computer-program in selected locations based on reducing power consumption and satisfying user-specified real-time performance constraints; and

wherein the power-down instruction in the selected locations reduce the power consumption during the execution of the code while satisfying the user-specified real-time performance constraints.

35. (Previously Presented) The medium of claim 34, wherein inserting power-down instructions in the computer-program in selected locations further comprises:

identifying one or more potential locations in the computer program where power-down instructions can be inserted;

selecting locations to insert the power-down instructions from the identified potential locations in the program based on satisfying user-specified real-time performance constraints; and inserting the power-down instructions in the selected locations to reduce power consumption while running the computer program while satisfying the user-specified real-time performance constraints.

36. (Original) The medium of claim 35, wherein the code is written for a microprocessor including distinct functional units.

37. (Original) The medium of claim 35, wherein identifying potential locations comprises:

identifying the potential locations based on the functional units not being used in the potential locations, wherein the functional units not being used are determined based on functional unit usage transfer functions at each of the potential locations as specified in standard monotone data-flow frameworks.

38. (Original) The medium of claim 37, wherein identifying potential locations is accomplished by statically analyzing processor cycles prior to running the program.

39. (Previously Presented) The medium of claim 35, wherein selecting the identified potential locations on the computer program based on satisfying the user-specified real-time constraints, comprise:

running the computer program to generate power-profiling information associated with each of the identified potential locations;

running the computer program to generate execution path-profiling information associated with each of the identified potential locations;

assigning a weight factor to each of the identified potential locations based on the generated power-profiling and path-profiling information; and

selecting the locations to insert the power-down instructions from the identified locations based on the assigned weight factors and the user-specified real-time performance constraints.

40. (Original) The medium of claim 39, wherein running the program to generate path-profiling information to each of the identified potential locations further comprises:

generating running probability of each of the identified potential locations based on the generated path-profiling information.

41. (Original) The medium of claim 40, wherein assigning the weight factor comprises:

extracting potential energy savings for each of the identified potential locations using the generated power profile analysis information; and

assigning the weight factor to each of the identified potential locations based on the extracted potential energy savings and the generated running probability.

42. (Original) The medium of claim 41, wherein assigning the weight factor further comprises:

running the program to assign a first weight factor based on the extracted potential energy savings to each of the identified potential locations;

running the program to assign a second weight factor based on execution probability at each of the identified potential locations;

computing a product of the first and second weight factors for each of the identified potential locations;

calculating the weight factor for each of the identified potential locations based on the computed product of the first and second weight factors; and

assigning the calculated weight factor to each of the identified potential locations.

43. (Previously Presented) The medium of claim 35, wherein user-specified real-time performance constraints comprise:

the number of power-down instructions that can be inserted in a running path including one or more identified potential locations.

44. (Original) The medium of claim 43, further comprising:

inserting power-up instructions in the program to restore at least one functional unit to a ready state powered-down by the inserted power-down instructions.